Version 1.9

User Guide

Merchant Swish Simulator

Date: 2020-06-09

Table of contents

1 Revision history

Date	Version	Name	Description
2015-11-05	0.9.8	AT	First release to publish
2015-12-10	0.9.8.1	CS	Renamed document to Guide Testverktyg 3.3 Added information about port. 4.2.1 Changed example callback url. 4.2.2 Changed example callback url. 4.3 Changed example callback url. 4.5 Deleted error codes AC05, AC06, AC07, AC15, AM04, AM14, AM21, and DS0K. Added error code RF07. 5.1 Changed example callback url. 5.2 Changed example callback url. 6.2 Changed example callback url. 6.3 Changed example callback url. 6.5 Deleted error codes AC05, AC06, AC07, AC15, AM04, AM14, AM21, and DS0K. Added error code RF07. Deleted note related to error code RF04. Added error code RF07. 7.1 Changed example callback url.
2016-01-18	1.0	PJ	Created version 1.0
2018-08-15	1.1	Magnus Lageson	Updated end-points
2018-10-24	1.2	Magnus Lageson	Updated Curl calls

2018-10-26	1.3	Mats Bergström	Document title: Renamed to ‘User Guide’. Ch. 2: Renamed from ‘Background’ to ‘Introduction’ and updated. Ch. 3.2: Added information about Technical Supplier certificate. Ch. 4.4; 7,4: Added information about MSS limitation for HTTP 403. Ch: 4.5; 7.5: Elaborated on how error codes can be used to simulate errors and added new codes applicable for Technical Supplier API user. Ch 7: Added information about MSS cache.
2019-02-18	1.4	Mats Bergström	Document updated with information about new optional parameters `payerSSN` and `ageLimit` for the Payment Request API. In addition all `curl` examples has been completely updated and verified. A new chapter regarding `curl` usage has been added and a number of old chapters has been renamed to better reflect their content as well updated with more information on how to use them.
2019-05-09	1.5	Per Nilsson	Added section on cancel payment request.

2019-11-27	1.6	Mats Bergström	Document updated with information related to payout request as well as some minor corrections and improvements to existing chapters.
2020-03-06	1.7	Mats Bergström	New chapters added: - Create payment request v2 (PUT) - Create refund request v2 (PUT) Minor corrections.
2020-04-06	1.8	Mats Bergström	Fix incorrect URL for Create refund request v2 (PUT)
2020-06-09	1.9	Mats Bergström	Add information about currently supported TLS version to chapter “TLS authentication certificate”.

2 Introduction

This document is intended for Merchants and Technical Suppliers (TS) who wish to verify their usage of the Swish Payment, Refund and Payout REST Application Programming Interfaces (API).

The Merchant Swish Simulator (MSS) is a stand-alone test environment that offer a way for merchants and technical suppliers to verify the format and content of API message sent to and received from Swish without any integration with other system components. An API call will return an error message or a correctly formatted response message depending on what data is put in the provided request data. This allows for the API user to validate both positive and negative (error) scenarios. Please note that contrary to the real Swish system the data communicated over these interfaces are completely separated from each other.

3 API usage

3.1 Endpoints simulated by MSS

MSS provides five endpoints (URL) for simulation of *Swish* merchant requests:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests`

`https://mss.cpc.getswish.net/swish-cpcapi/api/v2/paymentrequests/<id>`

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/refunds`

`https://mss.cpc.getswish.net/swish-cpcapi/api/v2/refunds/<id>`

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/payouts`

These are REST endpoints and the data to provide in the request data object (body) should be in JSON format.

Important to note is that contrary to the real Swish system there is no connection between these endpoints in MSS. They are completely independent of each other meaning that data created/retrieved through the paymentrequests endpoint are not available through the refunds endpoint.

3.2 TLS authentication certificates

In order to communicate with the *MSS server* the API client should use one of the provided client TLS certificate files:

- `Swish_Merchant_TestCertificate_1234679304.p12`
- `Swish_TechnicalSupplier_TestCertificate_9871065216.p12`

The password for the private key is *swish* for both certificates.

These files contains a Swish *test certificate* together with the complete chain of trust and the private key. The swish number specified in the file name is the Common Name (CN) used when the certificate was created.

The Swish Root CA certificate is available in the files mentioned above but also in a separate file “Swish_TLS_RootCA.pem”.

Using other certificates than the above will result in an error response (or no response at all). Likewise using other TLS version than v1.2 will cause TLS handshake error, v1.3 is not yet supported.

Normally, when a client sends a create payment request it must set the `payeeAlias` (swish number) data field to the number for which the certificate was created (Common Name property) or the request would be rejected.

Note that MSS will *not reject* any payment/refund requests due to mismatch between `payeeAlias` and certificate Common Name, any `payeeAlias` value can be used as long as it is a properly formatted swish number and the used authentication certificate is accepted by MSS. This also applies to the payout requests, MSS will *not reject* payout requests if the `payerAlias` property does not match common name in used authentication certificate.

3.3 Payload signing certificates

To be able to successfully process a payout request it is required that in addition to using a valid authentication certificate the payout request body must contain a valid `signature` property.

The signature property value should be the result of signing the *payout payload* using the payer *signing* private key which is different from the private key associated with the *authentication* certificate.

For this purpose a specific MSS *test signing key/certificate* is provided.

- `Swish_Merchant_TestSigningCertificate_1234679304.p12`

The password for the private key is: swish

The certificate serial number is: 7D70445EC8EF4D1E3A713427E973D097

The certificate common name (swish number) is: 1234679304

Important: *Since MSS is a stand-alone test environment it has no knowledge of any other signing certificates than the one stated above. Thus, for successful payout requests this certificate private key must be used for signing or MSS will reject the payout request since it will not be able to validate the signature.*

3.4 TLS for the callback endpoint

The client *callback endpoint* has to use HTTPS on port 443 and it is highly recommended to use IP filtering as well. For the callback MSS will be acting client and the merchant/technical supplier is acting server. Swish will validate the callback server TLS certificate against a list of commonly recognized Certificate Authorities (CA).

3.5 Call endpoints using `curl`

This document includes a number of examples on how one can manually call the API using `curl`, a command line tool for transferring data with URLs. Both positive and negative examples are provided.

To minimize the risk of running into `curl` tool related problems it is recommended to run the examples using a a fairly new version of `curl` built with *openSSL* in favour of *GnuTLS.* (the default version for Ubuntu 16.04 is 7.47.0 built with GnuTLS and will most likely not work but result in an error message: *curl: (35) error reading X.509 potentially-encrypted key file: Base64 decoding error.*)

For details on how to use `curl`, please refer to the man page.

All examples has been verified to work as expected when executed on a GNU/Linux machine with:

- Linux Kernel version: #61~16.04.1-Ubuntu SMP Wed Jun 14 11:58:22 UTC 2017
- curl version: curl 7.61.1 (x86_64-pc-linux-gnu) libcurl/7.61.1 OpenSSL/1.0.2g zlib/1.2.8

In the provided examples the used `curl` commands are built up like this:

```
$ curl -s -S -i --cert <path-to-certificate-file>:<password> --cert-type p12 --
cacert <path-to-rootCA-pem-file> --tlsv1.2 --header "Content-Type:
application/json" <endpoint-url> --data '<json-formatted-data>'
```

All examples are executed from the directory in the file system where the certificate files are located.

Note on copy-paste of example `curl` commands: Some PDF viewers insert newline characters on copy. Thus, if you want to copy an example command it might be necessary to first paste the copy into an editor and remove any newline characters before pasting into a shell.

4 Create payment request v1 (POST)

Merchants and Technical Suppliers can send create payment requests for both E-Commerce and M-Commerce to MSS.

Note: This POST request method is to be regarded as deprecated but is still supported. However, (new) users should preferably use the PUT request method described in chapter 5 since POST method will eventually be removed.

In order to send create payment request the request object needs to be a POST to URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests`

Once MSS receives a “create payment request” call, there are two answers that will be returned from MSS (unless error situation). The first answer is synchronous, the second one is asynchronous.

Please note the following:

1. MSS directly returns a “Payment request created” response to the caller.
   In case of M- Commerce the response will contain a *PaymentRequestToken* in the header in addition to the *Location* property. This token is unique for each payment request.
2. MSS sends a “Payment result” to the provided callback URL after some delay, normally four seconds. This delay is not configurable by the client.
   For the M-Commerce case the MSS server will set the `payerAlias` property to a fake value of “46464646464” in the response.
3. If the client simulates an error situation by providing an error code in the `message` property the error response could come immediately or in the callback depending on used error code.

4.1 Payment request data object

The Payment Request Object is used in all payment request operations and the provided data object (body) should be in JSON format. The table shows all possibly properties for a payment request, the properties marked as mandatory (M) must be present or the call will be rejected.

Property	Type	Mandatory/ Optional	Description
payeePaymentReference	string	O	Payment reference supplied by theMerchant. This is not used by Swish but is included in responses back to the client. This reference could for example be an order id or similar. If set the value must not exceed 35 characters and only the following characters are allowed: [a-ö, A-Ö, 0-9, -]

callbackUrl	string	M	URL that Swish will use to notify caller about the result of the payment request. The URL has to use HTTPS.
payerAlias	string	O	The registered Cell phone number of the person that makes the payment. It can only contain numbers and has to be at least 8 and at most 15 digits. It also needs to match the following format in order to be found in Swish: country code + cell phone number (without leading zero). E.g.: 46712345678 If set, request is handled as E-Commerce payment. If not set, request is handled as M-Commerce payment.
payerSSN	String	O	The social security number of the individual making the payment, should match the registered value for *payerAlias* or the payment will not be accepted. The value should be a proper Swedish social security number (personnummer or sammordningsnummer). Note: Since MSS is a stand-alone test system it can not verify if *payerSSN* match registered value for *payerAlias*. It is possible to simulate an 'ssn not matching' error, see *message* property below.
ageLimit	String	O	Minimum age (in years) that the individual connected to the *payerAlias* has to be in order for the payment to be accepted. Value has to be in the range of 1 to 99. Note: Since MSS is a stand-alone test system it can not verify the *payerAlias* age against the *ageLimit* value. It is possible to simulate an 'age to low' error, see *message* property below.
payeeAlias	string	M	The Swish number of the payee. It needs to match with Merchant Swish number.

amount	string	M	The amount of money to pay. The amount cannot be less than 0.01 SEK and not more than 999999999999.99 SEK. Valid value has to be all digits or with 2 digit decimal separated with a period.
currency	string	M	The currency to use. Currently the only supported value is SEK.
message	string	O	Merchant supplied message about the payment/order. Max 50 characters. Allowed characters are the letters a-ö, A-Ö, the numbers 0-9 and any of the special characters :;.,?!()-”. For MSS, an error code as defined in section 4.4 can be set in this *message* property in order to simulate an error situation.

4.2 Examples

4.2.1 Successful payment request, E-Commerce

A create payment request *with* the payer mobile telephone (`payerAlias`) number set is for E-Commerce.

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests
--data '{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payerAlias" : "4671234768",
"payeeAlias" : "1231181189", "amount" : "100", "currency" : "SEK", "message" :
"Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 201 Created
Location:
https://mss.cpc.getswish.net/swish-
cpcapi/v1/paymentrequests/AB23D7406ECE4542A80152D909EF9F6B

After some delay the provided callbackURL is called by MSS with the result of the
payment request.
```

4.2.2 Successful payment request, M-Commerce

A create payment request *without* the payer mobile telephone number (`payerAlias`) is for M-Commerce.

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests
--data '{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payeeAlias" : "1231181189", "amount" :
"100", "currency" : "SEK", "message" : "Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/98BF074EE6CA42F7BCBE175182D59659
Server: nginx/1.12.1
Connection: keep-alive
PaymentRequestToken: 00132ec0dda74b12acc142fa355181fc
Content-Length: 0
Date: Tue, 12 Feb 2019 14:22:21 GMT

After some delay the provided callbackURL is called by MSS with the result of the
payment request.
```

4.2.3 Error, invalid parameter

A create payment request with an invalid `payeeAlias` property.

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests
--data '{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payerAlias" : "4671234768",
"payeeAlias" : "9991181189", "amount" : "100", "currency" : "SEK", "message" :
"Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 403
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Tue, 12 Feb 2019 14:51:24 GMT

[{"errorCode":"PA01","errorMessage":"Parameter is not
correct.","additionalInformation":""}]
```

4.2.4 Simulating an error using message property

A create payment request with error code BE18 in `message` property, simulating an error situation. For a list of error codes refer to section 4.4.

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests
--data '{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payerAlias" : "4671234768",
"payeeAlias" : "1231181189", "amount" : "100", "currency" : "SEK", "message" :
"BE18" }'

HTTP/1.1 422 Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked Date: Tue, 12 Feb 2019 14:32:09 GMT

[{"errorCode":"BE18","errorMessage":"Payer alias is
invalid","additionalInformation":null}]
```

4.3 HTTP status codes

The following HTTP status codes could be received in a create payment response message:

HTTP status codes	Returned scenarios
201 Created	Returned when Payment request was successfully created. Will return a `Location` header. For the M-Commerce case the response will also contain a `PaymentRequestToken` header.
400 Bad Request	Returned when the Create Payment Request operation was malformed.
401 Unauthorized	Returned when there are authentication problems with the certificate. Or the Swish number in the certificate is not enrolled. Will return nothing else.
403 Forbidden	Returned when the `payeeAlias` in the payment request object is not the same as Merchants Swish number in the provided certificate or the value is not a valid swish number. Note: Since MSS is a test system, any mismatch between the swish number in provided certificate and the request data object are ignored by MSS, it is only logged as a mismatch in the server.
415 Unsupported Media Type	Returned when Content-Type header is not "application/json". Will return nothing else.
422 Unprocessable Entity	Returned when there are validation errors.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server, this should normally not happen. Will return nothing else.

4.4 Error Simulation codes

The client can trigger (simulate) an error situation by setting the `message` property of the *create payment request* data object to an appropriate value as listed in the following sections.

Depending on error code and type of payment (E-commerce/M-commerce) the simulated error will be triggered immediately in the first ‘Payment request created’ step or in the second delayed ‘Payment result’ step (callback).

4.4.1 Simulate failure for the ‘*Payment request created’* step

Simulate failure for the ‘*Payment request created’* step (first response):

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *422*. The included response object will contain the error code in its `errorCode` property and a description in the `errorMessage` property.

Error codes	Description
FF08	PayeePaymentReference is invalid
RP03	Callback URL is missing or does not use Https
BE18	Payer alias is invalid
RP01	Payee alias is missing or empty
PA02	Amount value is missing or not a valid number
AM06	Amount value is too low
AM02	Amount value is too large
AM03	Invalid or missing Currency
RP02	Wrong formatted message
RP06	Another active PaymentRequest already exists for this payerAlias. (Only applicable for E-Commerce.)
ACMT03	Payer not Enrolled
ACMT01	Counterpart is not activated
ACMT07	Payee not Enrolled
UNKW	Technical supplier is not active (only applicable for Technical Supplier API user).
VR01	Does not meet age limit. Note: Only E-Commerce case, for M-Commerce case the simulated error will trigger in the delayed 'Payment result callback' (second response).
VR02	SSN does not match enroled customer. Note: Only E-Commerce case, for M-Commerce case the simulated error will trigger in the delayed 'Payment result callback' (second response).

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *403*. The included response object will contain the error code in its `errorCode` property, no detailed information is provided.

Error codes	Description
PA01	The technical supplier is not connected to the merchant (only applicable for Technical Supplier API user).

4.4.2 Simulate failure for the '*Payment result*' step

Simulate failure for the '*Payment result*' step (second answer, data provided in callback):

Setting the `message` property to one of the following error codes will result in the `status` property of the *callback* data object to be set to `ERROR` and the `errorCode` property is set to the error code simulated.

Note: The simulated error situation is also indicated in any following GET requests sent to MSS, *refer to section 6.2 for an example.*

Error codes	Description
RF07	Transaction declined
BANKIDCL	Payer cancelled BankId signing
FF10	Bank system processing error
TM01	Swish timed out before the payment was started
DS24	Swish timed out waiting for an answer from the banks after payment was started. Note: If this happens Swish has no knowledge of whether the payment was successful or not. The Merchant should inform its consumer about this and recommend them to check with their bank about the status of this payment.
VR01	Does not meet age limit. Note: Only M-Commerce case, for E-Commerce case the simulated error will trigger in the 'Payment request created' step (first response).
VR02	SSN does not match enroled customer. Note: Only M-Commerce case, for E-Commerce case the simulated error will trigger in the 'Payment request created' step (first response).

5 Create payment request v2 (PUT)

Merchants and Technical Suppliers can send create payment requests for both E-Commerce and M-Commerce to MSS using PUT method.

Note: It is recommended to use this PUT request method in favor of the POST request method described in section 4. The POST method should be regarded as deprecated even though it is still supported.

In order to send create payment request the request object needs to be a PUT to URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v2/paymentrequests/<uuid>`

The `instructionUUID` <uuid> of the URL should be a unique identifier (UUID) created/generated by the caller and confirm to format ^[0-9A-F]{32}$, that is a 32 character hexadecimal number (upper case) represented as a string. The same `instructionUUID` will be used in the response message *Location* header property.

Once MSS receives a “create payment request” call, there are two answers that will be returned from MSS (unless error situation). The first answer is synchronous, the second one is asynchronous.

Please note the following:

1. MSS directly returns a “Payment request created” response to the caller.
   In case of M-Commerce the response will contain a *PaymentRequestToken* in the header in addition to the *Location* property. This token is unique for each payment request.

2. MSS sends a “Payment result” to the provided callback URL after some delay, normally four seconds. This delay is not configurable by the client.
   For the M-Commerce case the MSS server will set the `payerAlias` property to a fake value of “46464646464” in the response.

3. If the client simulates an error situation by providing an error code in the `message` property the error response could come immediately or in the callback depending on used error code.

5.1 Payment request data object

The Payment Request Object is used in all payment request operations and the provided data object (body) should be in JSON format. The table shows all possibly properties for a payment request, the properties marked as mandatory (M) must be present or the call will be rejected.

Property	Type	Mandatory/ Optional	Description
payeePaymentReference	string	O	Payment reference supplied by theMerchant. This is not used by Swish but is included in responses back to the client. This reference could for example be an order id or similar. If set the value must not exceed 35 characters and only the following characters are allowed: [a-ö, A-Ö, 0-9, -]
callbackUrl	string	M	URL that Swish will use to notify caller about the result of the payment request. The URL has to use HTTPS.
payerAlias	string	O	The registered Cell phone number of the person that makes the payment. It can only contain numbers and has to be at least 8 and at most 15 digits. It also needs to match the following format in order to be found in Swish: country code + cell phone number (without leading zero). E.g.: 46712345678 If set, request is handled as E-Commerce payment. If not set, request is handled as M-Commerce payment.

payerSSN	String	O	The social security number of the individual making the payment, should match the registered value for *payerAlias* or the payment will not be accepted. The value should be a proper Swedish social security number (personnummer or sammordningsnummer). Note: Since MSS is a stand-alone test system it can not verify if *payerSSN* match registered value for *payerAlias*. It is possible to simulate an 'ssn not matching' error, see *message* property below.
ageLimit	String	O	Minimum age (in years) that the individual connected to the *payerAlias* has to be in order for the payment to be accepted. Value has to be in the range of 1 to 99. Note: Since MSS is a stand-alone test system it can not verify the *payerAlias* age against the *ageLimit* value. It is possible to simulate an 'age to low' error, see *message* property below.
payeeAlias	string	M	The Swish number of the payee. It needs to match with Merchant Swish number.
amount	string	M	The amount of money to pay. The amount cannot be less than 0.01 SEK and not more than 999999999999.99 SEK. Valid value has to be all digits or with 2 digit decimal separated with a period.
currency	string	M	The currency to use. Currently the only supported value is SEK.

message	string	O	Merchant supplied message about the payment/order. Max 50 characters. Allowed characters are the letters a-ö, A-Ö, the numbers 0-9 and any of the special characters :;.,?!()-”. For MSS, an error code as defined in section 5.4 can be set in this *message* property in order to simulate an error situation.

5.2 Examples

5.2.1 Successful payment request, E-Commerce

A create payment request *with* the payer mobile telephone (`payerAlias`) number set is for E-Commerce.

```
$ curl -s -S -i -X PUT --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish
--cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net/swish-
cpcapi/api/v2/paymentrequests/2F9C2F35D92340348F130D702E6C4CCC --data
'{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payerAlias" : "4671234768",
"payeeAlias" : "1231181189", "amount" : "100", "currency" : "SEK", "message" :
"Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 201 Created
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/2F9C2F35D92340348F130D702E6C4CCC

After some delay the provided callbackURL is called by MSS with the result of the
payment request.
```

5.2.2 Successful payment request, M-Commerce

A create payment request *without* the payer mobile telephone number (`payerAlias`) is for M-Commerce.

```
$ curl -s -S -i -X PUT --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish
--cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-
cpcapi/api/v2/paymentrequests/2F9C2F35D92340348F130D702E6C4ACC --data
'{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payeeAlias" : "1231181189", "amount" :
"100", "currency" : "SEK", "message" : "Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/2F9C2F35D92340348F130D702E6C4ACC
Server: nginx/1.12.1
Connection: keep-alive
PaymentRequestToken: 00132ec0dda74b12acc142fa355181fc
Content-Length: 0
Date: Tue, 12 Feb 2019 14:22:21 GMT

After some delay the provided callbackURL is called by MSS with the result of the
payment request.
```

5.2.3 Error, invalid parameter

A create payment request with an invalid `payeeAlias` property.

```
$ curl -s -S -i -X PUT --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish
--cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-
cpcapi/api/v2/paymentrequests/2F9C2F35D92340348F130D702E6C4AAA --data
'{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payerAlias" : "4671234768",
"payeeAlias" : "9991181189", "amount" : "100", "currency" : "SEK", "message" :
"Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 403
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Tue, 12 Feb 2019 14:51:24 GMT

[{"errorCode":"PA01","errorMessage":"Parameter is not
correct.","additionalInformation":""}]
```

5.2.4 Simulating an error using message property

A create payment request with error code BE18 in `message` property, simulating an error situation. For a list of error codes refer to section 5.4.

```
$ curl -s -S -i -X PUT --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish
--cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-
cpcapi/api/v2/paymentrequests/2F9C2F35D92340348F130D702E6C4AAB --data
'{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payerAlias" : "4671234768",
"payeeAlias" : "1231181189", "amount" : "100", "currency" : "SEK", "message" :
"BE18" }'

HTTP/1.1 422 Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked Date: Tue, 12 Feb 2019 14:32:09 GMT

[{"errorCode":"BE18","errorMessage":"Payer alias is
invalid","additionalInformation":null}]
```

5.3 HTTP status codes

The following HTTP status codes could be received in a create payment response message:

HTTP status codes	Returned scenarios
201 Created	Returned when Payment request was successfully created. Will return a `Location` header. For the M-Commerce case the response will also contain a `PaymentRequestToken` header.
400 Bad Request	Returned when the Create Payment Request operation was malformed.
401 Unauthorized	Returned when there are authentication problems with the certificate. Or the Swish number in the certificate is not enrolled. Will return nothing else.
403 Forbidden	Returned when the `payeeAlias` in the payment request object is not the same as Merchants Swish number in the provided certificate or the value is not a valid swish number. Note: Since MSS is a test system, any mismatch between the swish number in provided certificate and the request data object are ignored by MSS, it is only logged as a mismatch in the server.
415 Unsupported Media Type	Returned when Content-Type header is not "application/json". Will return nothing else.
422 Unprocessable Entity	Returned when there are validation errors.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server, this should normally not happen. Will return nothing else.

5.4 Error Simulation codes

The client can trigger (simulate) an error situation by setting the `message` property of the *create payment request* data object to an appropriate value as listed in the following sections.

Depending on error code and type of payment (E-commerce/M-commerce) the simulated error will be triggered immediately in the first 'Payment request created' step or in the second delayed 'Payment result' step (callback).

5.4.1 Simulate failure for the '*Payment request created*' step

Simulate failure for the '*Payment request created*' step (first response):

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *422*. The included response object will contain the error code in its `errorCode` property and a description in the `errorMessage` property.

Error codes	Description
FF08	PayeePaymentReference is invalid
RP03	Callback URL is missing or does not use Https
BE18	Payer alias is invalid
RP01	Payee alias is missing or empty
PA02	Amount value is missing or not a valid number
AM06	Amount value is too low
AM02	Amount value is too large
AM03	Invalid or missing Currency
RP02	Wrong formatted message
RP06	Another active PaymentRequest already exists for this payerAlias. (Only applicable for E-Commerce.)
RP09	The given instructionUUID is not available Note: The `instructionUUID` already exist in the database, i.e. it is not unique.
ACMT03	Payer not Enrolled
ACMT01	Counterpart is not activated
ACMT07	Payee not Enrolled
UNKW	Technical supplier is not active (only applicable for Technical Supplier API user).

VR01	Does not meet age limit. Note: Only E-Commerce case, for M-Commerce case the simulated error will trigger in the delayed 'Payment result callback' (second response).
VR02	SSN does not match enroled customer. Note: Only E-Commerce case, for M-Commerce case the simulated error will trigger in the delayed 'Payment result callback' (second response).

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *403*. The included response object will contain the error code in its `errorCode` property, no detailed information is provided.

Error codes	Description
PA01	The technical supplier is not connected to the merchant (only applicable for Technical Supplier API user).

5.4.2 Simulate failure for the '*Payment result*' step

Simulate failure for the '*Payment result*' step (second answer, data provided in callback):

Setting the `message` property to one of the following error codes will result in the `status` property of the *callback* data object to be set to `ERROR` and the `errorCode` property is set to the error code simulated.

Note: The simulated error situation is also indicated in any following GET requests sent to MSS, *refer to section 6.2 for an example.*

Error codes	Description
RF07	Transaction declined
BANKIDCL	Payer cancelled BankId signing
FF10	Bank system processing error
TM01	Swish timed out before the payment was started
DS24	Swish timed out waiting for an answer from the banks after payment was started. Note: If this happens Swish has no knowledge of whether the payment was successful or not. The Merchant should inform its consumer about this and recommend them to check with their bank about the status of this payment.

VR01	Does not meet age limit. Note: Only M-Commerce case, for E-Commerce case the simulated error will trigger in the ‘Payment request created’ step (first response).
VR02	SSN does not match enroled customer. Note: Only M-Commerce case, for E-Commerce case the simulated error will trigger in the ‘Payment request created’ step (first response).

6 Retrieve payment result (GET)

The client can retrieve payment result information of an initiated payment request (successful or failed) by sending a GET request to the following URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests/<id>`

The complete URL (with the payment request identifier) can be found in the HTTP `Location` header returned in the response message for a previous *create payment request* call. Alternatively, if the create payment request PUT method was used (section 5) the `id` is the `instructionUUID` provided in that request.

If the previous *create payment request* call simulated a delayed error the response of the GET operation will have a `status` property with value ERROR and properties `errorCode` and `errorMessage` will be set accordingly. See chapter 4.4.2/5.4.2 for possible errors that can be simulated with a delay.

For complete examples of positive and negative cases please refer to sections 6.1 and 6.2.

Remark: MSS stores the necessary information about each incoming “Payment request” in a cache which automatically expires every 24 hours or when the MSS server is restarted.

6.1 Example, Retrieve payment request result

Create payment request:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json"  https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests
--data '{ "payeePaymentReference" : "0123456789", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "payeeAlias" : "1231181189", "amount" :
"100", "currency" : "SEK", "message" : "Kingston USB Flash Drive 8 GB" }'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/98BF074EE6CA42F7BCBE175182D59659
Server: nginx/1.12.1
Connection: keep-alive
PaymentRequestToken: 00132ec0dda74b12acc142fa355181fc
Content-Length: 0
Date: Tue, 12 Feb 2019 14:22:21 GMT
```

After delay time, Retrieve the payment result:

```
curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-
type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/98BF074EE6CA42F7BCBE175182D59659

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Tue, 12 Feb 2019 15:33:34 GMT

{"id":"98BF074EE6CA42F7BCBE175182D59659", "payeePaymentReference":"0123456789",
"paymentReference":"6E7DBB1C56CB43E787CBB18F906A585D",
"callbackUrl":"https://myfakehost.se/swishcallback.cfm",
"payerAlias":"46464646464", "payeeAlias":"1231181189", "amount":100.00,
"currency":"SEK", "message":"Kingston USB Flash Drive 8 GB", "status":"PAID",
"dateCreated":"2019-02-12T14:22:21.610Z", "datePaid":"2019-02-12T14:22:25.610Z",
"errorCode":null, "errorMessage":null}
```

6.2 Example. Retrieve payment with simulated error

Create payment request with a simulated delayed error (M-commerce, VR01):

```
curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type: application/json" https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests --data '{ "payeePaymentReference" : "0123456789", "callbackUrl" : "https://myfakehost.se/swishcallback.cfm",  "payeeAlias" : "1231181189", "amount" : "100", "currency" : "SEK", "ageLimit": "18", "message" : "VR01" }'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests/BC4C3F91A3FD44D58F97D0A11C08EA34
Server: nginx/1.12.1
Connection: keep-alive
PaymentRequestToken: ff4937140f214daaa7c946aaa8cd09f8
Content-Length: 0
Date: Tue, 12 Feb 2019 16:11:46 GMT
```

After delay time, Retrieve the payment result:

```
curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type: application/json" https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests/BC4C3F91A3FD44D58F97D0A11C08EA34

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Tue, 12 Feb 2019 16:12:30 GMT
{"id":"BC4C3F91A3FD44D58F97D0A11C08EA34", "payeePaymentReference":"0123456789", "paymentReference":"CC3F94D57A54451DA0F2CEDFD89C24FE", "callbackUrl":"https://myfakehost.se/swishcallback.cfm", "payerAlias":"46464646464","payeeAlias":"1231181189","amount":100.00, "currency":"SEK", "message":"VR01", "status":"ERROR", "dateCreated":"2019-02-12T16:11:46.040Z", "datePaid":null, "errorCode":"VR01", "errorMessage":"Does not meet age limit"}
```

6.3 HTTP status codes

The following HTTP status codes could be received in a retrieve payment result response:

HTTP status codes	Returned scenarios
200 OK	Returned when Payment request was found. Will return Payment Request Object. The *status* property will indicate if the payment was successful or not.
401 Unauthorized	Returned when there are authentication problems with the certificate. Will return nothing else.
404 Not found	Returned when the Payment request was not found. Will return nothing else.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server, this should normally not happen. Will return nothing else.

7 Cancel Payment Request (PATCH)

A payment request can be cancelled by the client with a Http PATCH call to the payment request uri (same uri as for retrieve with GET) with content specifying changing status to cancelled:

```
https://mss.cpc.getswish.net/swish-cpcapi/api/v1/paymentrequests/<id>
```

A request can be cancelled until it is in status PAID, DECLINED, ERROR or CANCELLED.

On success, the response will be the same as for retrieve (GET), with status CANCELLED and a payment result is posted to the provided payment callback url.

The CANCELLED status is final. No more changes can be made to the request.

If there is an error, eg the payment request can not be cancelled due to already being cancelled, a response with http status 422 and json content with a swish error code is returned.

7.1 Cancel payment request data object

A PATCH request contains content specifying a list of changes on a resource. The only change allowed on a payment request is cancel, so cancel will be a list of one change. The change format is Json Patch (RFC6902) that defines a list of operation objects in Json. The cancelation operation has three attributes, with fixed values:

- op – the operation to perform, ie “replace”
- path – the search path of the attribute of target payment request object replace, ie “/status”
- value – what value to apply to the to the path, ie “cancelled”

Thus, the cancel payment request content will always be:

```
[{“op”: “replace”, “path”: “/status”, “value”: “cancelled”}]
```

The MIME type for Json Patch content is “application/json-patch+json”. This must be set in the Content-Type header.

7.2 Example cancel payment request

Example cancel payment request:

```
curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-
type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2
-X PATCH
--header "Content-Type: application/json-patch+json"
--data '[{"op": "replace", "path": "/status", "value": "cancelled"}]'
https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/98BF074EE6CA42F7BCBE175182D59659

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Tue, 12 Feb 2019 15:33:34 GMT

{"id":"98BF074EE6CA42F7BCBE175182D59659", "payeePaymentReference":"0123456789",
"paymentReference":"6E7DBB1C56CB43E787CBB18F906A585D",
"callbackUrl":"https://myfakehost.se/swishcallback.cfm",
"payerAlias":"46464646464", "payeeAlias":"1231181189", "amount":100.00,
"currency":"SEK", "message":"Kingston USB Flash Drive 8 GB", "status":"CANCELLED",
"dateCreated":"2019-02-12T14:22:21.610Z", "datePaid":"2019-02-12T14:22:25.610Z",
"errorCode":null, "errorMessage":null}
```

7.3 Example cancel payment request, error

Example cancel payment request error:

```
curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-
type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2
-X PATCH
--header "Content-Type: application/json-patch+json"
--data '[{"op": "replace", "path": "/status", "value": "cancelled"}]'
https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/paymentrequests/98BF074EE6CA42F7BCBE175182D59659

HTTP/1.1 422
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Mon, 18 Feb 2019 16:12:33 GMT

[{"errorCode":"RP07","errorMessage":"The payment request can not be
cancelled.","additionalInformation":null}]
```

7.4 HTTP Status Codes

HTTP status codes	Returned scenarios
200 OK	Returned when Payment request successfully cancelled.
401 Unauthorized	Returned when there are authentication problems with the certificate. Will return nothing else.
404 Not found	Returned when the Payment request was not found. Will return nothing else.
415 Unsupported Media Type	Returned if the Content-Type header is not “application/json-patch+json”.
422 Unprocessable entity	The cancellation could not be executed. See content for swish error code: RP07 – payment request is not in a cancellable state, PA01 – the cancel request operation submitted is invalid, only [{“op”: “replace”, “path”:”/status”, “value”:”cancelled”}] is allowed.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server, this should normally not happen. Will return nothing else.

8 Create refund request v1 (POST)

Merchants and Technical Suppliers can send create refund request to MSS.

Note: This POST request method is to be regarded as deprecated but is still supported. However, (new) users should preferably use the PUT request method described in chapter 9 since POST method will eventually be removed.

In order to send create refund request the request object needs to be POST to URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/refunds/`

When MSS receives a “Refund request” there are three answers that will be returned from MSS (unless error situation). The first answer is synchronous, the second and third responses are asynchronous.

Please note the following:

1. Contrary to the real Swish system there is no connection between the ‘Create payment request’ and ‘Create refund request’ flows in MSS. Thus, MSS can not validate that the values for `originalPaymentRequest`, nor `amount` are valid.
   The user can however simulate errors by setting the message property to appropriate error code. Refer to section 8.5 for possible error codes
2. MSS directly returns a “Refund request created” response to the caller.
3. MSS sends an intermediate “Refund result” (status DEBITED) to the provided callback URL after some delay, normally four seconds. The delay is not configurable by the client.
4. MSS sends a final “Refund result” (status PAID) to the provided callback URL after a second delay, normally four seconds. The delay is not configurable by the client.

8.1 Refund request object

The Refund request object is used in all refund operations and the provided data object (body) should be in JSON format. The table shows all possibly properties for a payment request, the properties marked as mandatory (M) must be present or the call will be rejected.

Property	Type	Mandatory/ Optional	Description
payerPaymentReference	string	O	Payment reference supplied by the Merchant. This is not used by Swish but is included in responses back to the client.

originalPaymentReference	string	M	Payment reference to the original payment that this refund is for.
callbackUrl	string	M	URL that Swish will use to notify caller about the outcome of the refund. The URL has to use HTTPS.
payerAlias	string	M	The Swish number of the Merchant that makes the refund payment.
payeeAlias	string	O	The Cell phone number of the person that receives the refund payment.
amount	string	M	The amount of money to refund. The amount cannot be less than 0.01 SEK and not more than 999999999999.99 SEK. The amount cannot exceed the remaining amount of the original payment that the refund is for. Note that MSS actually do not check if the refund is greater than the remaining amount of the original payment.
currency	string	M	The currency to use. Only supported value currently is SEK.
message	string	O	Merchant supplied message about the refund. Max 50 chars. Allowed characters are the letters a-o¨, A-Ö, the numbers 0-9 and the special characters :;.,?!()”. For MSS, an error code as defined in section 8.5 can be set in this message property in order to simulate an error situation.

8.2 Example, Successful refund request

Create refund request:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net:443/swish-cpcapi/api/v1/refunds --
data '{ "payerPaymentReference" : "0123456789", "originalPaymentReference" :
"6D6CD7406ECE4542A80152D909EF9F6B", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "amount" : "100", "currency" : "SEK",
"payerAlias" : "1234567839", "payeeAlias" : "9991234569", "message": "Refund for
Kingston SSD Drive 320 GB"}'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/refunds/D77BE41AF953468CADCA21D244724941
Server: nginx/1.12.1
Connection: keep-alive
Content-Length: 0D
Date: Mon, 18 Feb 2019 16:00:05 GMT

After some delay the provided callbackURL is called by MSS twice (unless error
situation) with the result of the payment request.
```

8.3 Example, Simulating an error using message property

A “Refund request” call with error code ACMT07 in `message` property simulating an error situation.

Create refund request:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net:443/swish-cpcapi/api/v1/refunds --
data '{ "payerPaymentReference" : "0123456789", "originalPaymentReference" :
"6D6CD7406ECE4542A80152D909EF9F6B", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "amount" : "100", "currency" : "SEK",
"payerAlias" : "1234567839", "payeeAlias" : "9991234569", "message": "ACMT07"}'

HTTP/1.1 422
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Mon, 18 Feb 2019 16:12:33 GMT

[{"errorCode":"ACMT07","errorMessage":"Payee alias not
enrolled","additionalInformation":null}]
```

8.4 HTTP status codes

The following HTTP status codes could be received in a create refund response:

HTTP status codes	Returned scenarios
201 Created	Returned when Refund was successfully created. Will return a `Location` header.
400 Bad Request	Returned when Create refund POST operation was malformed.
401 Unauthorized	Returned when there are authentication problems with the certificate. Or the Swish number in the certificate is not enrolled. Will return nothing else.
403 Forbidden	Returned when the `payerAlias` in the refund object is not the same as Merchants Swish number in the provided certificate. Note: This is not the case for MSS. Any mismatch between the swish number in provided certificate and the request data object are ignored by MSS, it is only logged as a mismatch in the MSS server. The only case where 403 is returned is if the API user has deliberately simulated an error resulting in 403.
415 Unsupported Media Type	Returned when Content-Type header is not "application/json". Will return nothing else.
422 Unprocessable Entity	Returned when there are validation errors. Will return an Array of Error Objects.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server, this should normally not happen. Will return nothing else.
504 Gateway Timeout	Returned when the Bank validation answers take too long and Swish times out. This rarely happens.

8.5 Error simulation codes

The API user can trigger (simulate) an error situation by setting the `message` property of the *Create refund request* data object to appropriate value as indicated in the following tables.

Simulate failure for the '*Refund response*' step (first response):

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *422*. The included response object will contain the error code in its `errorCode` property and the description in the `errorMessage` property.

Error codes	Description
FF08	PayerPaymentReference is invalid
RP03	Callback URL is missing or does not use Https
PA02	Amount value is missing or not a valid number
AM06	Amount value is too low
RF08	Amount value is too large or amount exceeds the amount of the original payment minus any previous refunds. Note: the remaining available amount is put into the additional information field.
AM03	Invalid or missing Currency
RP01	Payer alias is missing or empty
RP02	Invalid Message text
ACMT07	Payee not Enrolled
ACMT01	Counterpart is not activated
RF02	Original Payment not found or original payment is more than than 13 months old
RF03	Payer alias in the refund does not match the payee alias in the original payment.
RF04	Payer organization number does not match original payment payee organization number.
RF06	The Payee SSN (personnummer) in the original payment is not the same as the SSN for the current Payee. Note: Typically this means that the Mobile number has been transferred to another person.
BE18	Invalid contact details error
UNKW	Technical supplier is not active (only applicable for Technical Supplier API user).

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *403*. The included response object will contain the error code in its `errorCode` property.

Error codes	Description
PA01	The technical supplier is not connected to the merchant (only applicable for Technical Supplier API user).

Simulate failure for the '*Refund result*' step (second answer, data provided in callback):

Setting the *message* property to one of the following error codes will result in the `status` property of the *callback* data object will be set to `ERROR` and the `errorCode` property is set to the error code simulated.

Error codes	Description
RF07	Transaction declined
BANKIDCL	Payer cancelled BankId signing
FF10	Bank system processing error
DS24	Swish timed out waiting for an answer from the banks after payment was started. Note: If this happens Swish has no knowledge of whether the payment was successful or not. The Merchant should inform its consumer about this and recommend them to check with their bank about the status of this payment.

9 Create refund request v2 (PUT)

Merchants and Technical Suppliers can send create refund request to MSS using PUT method.

Note: It is recommended to use this PUT request method in favor of the POST request method described in section 8. The POST method should be regarded as deprecated even though it is still supported.

In order to send create refund request the request needs to be PUT to URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v2/refunds/<instructionUUID>`

The `instructionUUID` of the URL should be a unique identifier (UUID) created/generated by the caller and confirm to format ^[0-9A-F]{32}$, that is a 32 character hexadecimal number (upper case) represented as a string. The same `instructionUUID` will be used in the response message *Location* header property.

When MSS receives a “Refund request” there are three answers that will be returned from MSS (unless error situation). The first answer is synchronous, the second and third responses are asynchronous.

Please note the following:

5. Contrary to the real Swish system there is no connection between the ‘Create payment request’ and ‘Create refund request’ flows in MSS. Thus, MSS can not validate that the values for `originalPaymentRequest`, nor `amount` are valid.
   The user can however simulate errors by setting the message property to appropriate error code. Refer to section 9.5 for possible error codes
6. MSS directly returns a “Refund request created” response to the caller.
7. MSS sends an intermediate “Refund result” (status DEBITED) to the provided callback URL after some delay, normally four seconds. The delay is not configurable by the client.
8. MSS sends a final “Refund result” (status PAID) to the provided callback URL after a second delay, normally four seconds. The delay is not configurable by the client.

9.1 Refund request object

The Refund request object is used in all refund operations and the provided data object (body) should be in JSON format. The table shows all possibly properties for a payment request, the properties marked as mandatory (M) must be present or the call will be rejected.

Property	Type	Mandatory/ Optional	Description
payerPaymentReference	string	O	Payment reference supplied by the Merchant. This is not used by Swish but is included in responses back to the client.

originalPaymentReference	string	M	Payment reference to the original payment that this refund is for.
callbackUrl	string	M	URL that Swish will use to notify caller about the outcome of the refund. The URL has to use HTTPS.
payerAlias	string	M	The Swish number of the Merchant that makes the refund payment.
payeeAlias	string	O	The Cell phone number of the person that receives the refund payment.
amount	string	M	The amount of money to refund. The amount cannot be less than 0.01 SEK and not more than 999999999999.99 SEK. The amount cannot exceed the remaining amount of the original payment that the refund is for. Note that MSS actually do not check if the refund is greater than the remaining amount of the original payment.
currency	string	M	The currency to use. Only supported value currently is SEK.
message	string	O	Merchant supplied message about the refund. Max 50 chars. Allowed characters are the letters a-o¨, A-Ö, the numbers 0-9 and the special characters :;.,?!()”. For MSS, an error code as defined in section 9.5 can be set in this message property in order to simulate an error situation.

9.2 Example, Successful refund request

Create refund request:

```
$ curl -s -S -i -X PUT --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish
--cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net:443/swish-cpcapi/api/v1/refunds --
data '{ "payerPaymentReference" : "0123456789", "originalPaymentReference" :
"6D6CD7406ECE4542A80152D909EF9F6B", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "amount" : "100", "currency" : "SEK",
"payerAlias" : "1234567839", "payeeAlias" : "9991234569", "message": "Refund for
Kingston SSD Drive 320 GB"}'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/refunds/D77BE41AF953468CADCA21D244724941
Server: nginx/1.12.1
Connection: keep-alive
Content-Length: 0D
Date: Mon, 18 Feb 2019 16:00:05 GMT

After some delay the provided callbackURL is called by MSS twice (unless error
situation) with the result of the payment request.
```

9.3 Example, Simulating an error using message property

A “Refund request” call with error code ACMT07 in `message` property simulating an error situation.

Create refund request:

```
$ curl -s -S -i -X PUT --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish
--cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net:443/swish-cpcapi/api/v1/refunds --
data '{ "payerPaymentReference" : "0123456789", "originalPaymentReference" :
"6D6CD7406ECE4542A80152D909EF9F6B", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "amount" : "100", "currency" : "SEK",
"payerAlias" : "1234567839", "payeeAlias" : "9991234569", "message": "ACMT07"}'

HTTP/1.1 422
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Mon, 18 Feb 2019 16:12:33 GMT

[{"errorCode":"ACMT07","errorMessage":"Payee alias not
enrolled","additionalInformation":null}]
```

9.4 HTTP status codes

The following HTTP status codes could be received in a create refund response:

HTTP status codes	Returned scenarios
201 Created	Returned when Refund was successfully created. Will return a `Location` header.
400 Bad Request	Returned when Create refund POST operation was malformed.
401 Unauthorized	Returned when there are authentication problems with the certificate. Or the Swish number in the certificate is not enrolled. Will return nothing else.
403 Forbidden	Returned when the `payerAlias` in the refund object is not the same as Merchants Swish number in the provided certificate. Note: This is not the case for MSS. Any mismatch between the swish number in provided certificate and the request data object are ignored by MSS, it is only logged as a mismatch in the MSS server. The only case where 403 is returned is if the API user has deliberately simulated an error resulting in 403.
415 Unsupported Media Type	Returned when Content-Type header is not "application/json". Will return nothing else.
422 Unprocessable Entity	Returned when there are validation errors. Will return an Array of Error Objects.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server, this should normally not happen. Will return nothing else.
504 Gateway Timeout	Returned when the Bank validation answers take too long and Swish times out. This rarely happens.

9.5 Error simulation codes

The API user can trigger (simulate) an error situation by setting the `message` property of the *Create refund request* data object to appropriate value as indicated in the following tables.

Simulate failure for the '*Refund response*' step (first response):

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *422*. The included response object will contain the error code in its `errorCode` property and the description in the `errorMessage` property.

Error codes	Description
FF08	PayerPaymentReference is invalid
RP03	Callback URL is missing or does not use Https
PA02	Amount value is missing or not a valid number
AM06	Amount value is too low
RF08	Amount value is too large or amount exceeds the amount of the original payment minus any previous refunds. Note: the remaining available amount is put into the additional information field.
AM03	Invalid or missing Currency
RP01	Payer alias is missing or empty
RP02	Invalid Message text
ACMT07	Payee not Enrolled
ACMT01	Counterpart is not activated
RF02	Original Payment not found or original payment is more than than 13 months old
RF03	Payer alias in the refund does not match the payee alias in the original payment.
RF04	Payer organization number does not match original payment payee organization number.
RF06	The Payee SSN (personnummer) in the original payment is not the same as the SSN for the current Payee. Note: Typically this means that the Mobile number has been transferred to another person.
RF09	A refund with the given instructionUUID is already in progress Note: For MSS this error can only be simulated using `message` property.
RF09	The given instructionUUID is not available Note: The `instructionUUID` already exist in the database, i.e. it is not unique.

BE18	Invalid contact details error
UNKW	Technical supplier is not active (only applicable for Technical Supplier API user).

Setting the *message* property to one of the following error codes will result in an error response message with HTTP status code *403*. The included response object will contain the error code in its `errorCode` property.

Error codes	Description
PA01	The technical supplier is not connected to the merchant (only applicable for Technical Supplier API user).

Simulate failure for the '*Refund result*' step (second answer, data provided in callback):

Setting the *message* property to one of the following error codes will result in the `status` property of the *callback* data object will be set to `ERROR` and the `errorCode` property is set to the error code simulated.

Error codes	Description
RF07	Transaction declined
BANKIDCL	Payer cancelled BankId signing
FF10	Bank system processing error
DS24	Swish timed out waiting for an answer from the banks after payment was started. Note: If this happens Swish has no knowledge of whether the payment was successful or not. The Merchant should inform its consumer about this and recommend them to check with their bank about the status of this payment.

10 Retrieve refund result (GET)

The client can retrieve refund result information of an initiated refund request (successful or failed) by sending a GET request to the following URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/refunds/<id>`

The complete URL (with the refund request identifier) can be found in the HTTP `Location` header returned in the response message for a previous *create refund request* call.

If the previous *create refund request* call simulated a delayed error the response of the GET operation will have a `status` property with value ERROR and properties `errorCode` and `errorMessage` will be set accordingly. See chapter 7.5 for possible errors that can be simulated with a delay.

Remark: MSS stores the necessary information about each incoming “Refund request” in a cache which automatically expires every 24 hours or when the MSS server is restarted.

10.1 Example, Retrieve refund result

Create refund request:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net:443/swish-cpcapi/api/v1/refunds --
data '{ "payerPaymentReference" : "0123456789", "originalPaymentReference" :
"6D6CD7406ECE4542A80152D909EF9F6B", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "amount" : "100", "currency" : "SEK",
"payerAlias" : "1234567839", "payeeAlias" : "9991234569", "message": "Refund for
Kingston SSD Drive 320 GB"}'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/refunds/D77BE41AF953468CADCA21D244724941
Server: nginx/1.12.1
Connection: keep-alive
Content-Length: 0D
Date: Mon, 18 Feb 2019 16:00:05 GMT
```

After delay time, Retrieve refund result:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/refunds/D77BE41AF953468CADCA21D244724941

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Mon, 18 Feb 2019 17:34:43 GMT
{"id":"D77BE41AF953468CADCA21D244724941",
"paymentReference":"20CC72A882A64488ABC0878C623043EF",
"payerPaymentReference":"0123456789",
"originalPaymentReference":"6D6CD7406ECE4542A80152D909EF9F6B",
"callbackUrl":"https://myfakehost.se/swishcallback.cfm", "payerAlias":"1234567839",
"payeeAlias":"9991234569", "amount":100.00, "currency":"SEK", "message":"Refund for
Kingston SSD Drive 320 GB", "status":"PAID", "dateCreated":"2019-02-
18T16:00:05.484Z", "datePaid":"2019-02-18T16:00:13.605Z", "errorMessage":null,
"additionalInformation":null, "errorCode":null}
```

10.2 Example, Retrieve refund with simulated error

Create refund request with delayed simulated error:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net:443/swish-cpcapi/api/v1/refunds --
data '{ "payerPaymentReference" : "0123456789", "originalPaymentReference" :
"6D6CD7406ECE4542A80152D909EF9F6B", "callbackUrl" :
"https://myfakehost.se/swishcallback.cfm", "amount" : "100", "currency" : "SEK",
"payerAlias" : "1234567839", "payeeAlias" : "9991234569", "message": "DS24"}'

HTTP/1.1 201
Location: https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/refunds/DB68078911964A73A60D0D0507FACECF
Server: nginx/1.12.1
Connection: keep-alive
Content-Length: 0
Date: Mon, 18 Feb 2019 17:48:46 GMT
```

After delay time, Retrieve refund result:

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/refunds/DB68078911964A73A60D0D0507FACECF

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Mon, 18 Feb 2019 17:50:26 GMT

{"id":"DB68078911964A73A60D0D0507FACECF","paymentReference":null,"payerPaymentRefer
ence":"0123456789","originalPaymentReference":"6D6CD7406ECE4542A80152D909EF9F6B","c
allbackUrl":"https://myfakehost.se/swishcallback.cfm","payerAlias":"1234567839","pa
yeeAlias":"9991234569","amount":100.00,"currency":"SEK","message":"DS24","status":"
ERROR","dateCreated":"2019-02-
18T17:48:46.264Z","datePaid":null,"errorMessage":"Swish timed out waiting for an
answer from the banks after payment was
started","additionalInformation":null,"errorCode":"DS24"
```

10.3 HTTP status codes

The following HTTP status codes could be received in a retrieve refund result response:

HTTP status codes	Returned scenarios
200 OK	Returned when refund was found. Will return Refund Object..
401 Unauthorized	Returned when there are authentication problems with the certificate. Will return nothing else.
404 Not found	Returned when no refund was found or it was not created by the Merchant. Will return nothing else. Note that for MSS there is no check that the refund was created by the merchant that tries to retrieve the refund result. As long as the refund exist the result will be returned to the calling client.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server. Will return nothing else.

11 Create payout request (POST)

Merchants and Technical Suppliers can send create payout requests to MSS.

The request should to be of method POST and sent to URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/payouts`

When MSS receives the create payout request, at least one response will be returned to the caller depending on the provided data and the result of the request processing.

Please note the following:

1. MSS first returns a synchronous response to the caller, either a successful “payout request created” response (http status: 201) or an error message. A successful response consists of an empty body and a `Location` header containing a link to the specific payout instruction created.

2. If no `callbackURL` property was provided in the request it is the responsibility of the caller to check the status of the payout instruction using GET request as described in chapter 11.

3. If the `callbackURL` property was set in the request and the first synchronous response was successful MSS will send payout status information messages to the provided callback URL. The callback is sent after some delay (not configurable by the client). The body of the callback contains among other data the current status of the request. The provided data is the same as for the response to a GET request (refer to chapter 11.1 for examples).

   a) The first callback is sent after a delay of approximately four seconds and will contain status=DEBITED (unless an error situation).

   b) The second callback is sent after another delay of approximately four seconds and will containing final status=PAID (unless an error situation).

4. If the client simulates an error situation by providing a specific error simulation code in the `message` property (refer to chapter 93) an error response could come immediately or the error is indicated the callback depending on used error simulation code.

11.1 Payout request body

The payout request body should be a string in JSON format and contain two mandatory properties, payload and signature and the optional callbackURL:

```
{ "payload" : {...}, "callbackUrl" : "...", "signature" : "..." }
```

- `payload`:
  The payload property consists of value pairs following the JSON format and carries data about the payout to be executed. Most but not all payload properties are mandatory as described in the properties table below.
- `signature`:
  In order to verify the integrity of the payload data, a signature of the payload must be supplied. How to created the signature is outlined in chapter 10.2.

Below is an example on how a complete payout request body could look like (the signature property has been stripped down to save space).

```
{
    "payload": {
        "amount": "100.00",
        "currency": "SEK",
        "instructionDate": "2019-12-02T12:41:20",
        "message": "example message",
        "payeeAlias": "0728648607",
        "payeeSSN": "196702234292",
        "payerAlias": "1234679304",
        "payerPaymentReference": "mockedPayerPaymentReference",
        "payoutInstructionUUID": "822735C159A944DBB2B3546717BC919F",
        "payoutType": "PAYOUT",
        "signingCertificateSerialNumber": "7D70445EC8EF4D1E3A713427E973D097"
    },
    "callbackUrl": "https://myfavoritesite.dummy.domain",
    "signature": "IUybofWGga+uS2+APequDO5kh..."
}
```

The following table lists all possibly properties for a payout request, those marked as mandatory must be present or the call will be rejected with an error message.

Property	Type	Mandatory/ Optional	Format and Description
Payload Properties			
payoutInstructionUUID	String	Mandatory	A UUID of length 32. All **upper** case hexadecimal characters. A unique identifier created by the merchant to uniquely identify a payout instruction sent to the Swish system. Swish uses this identifier to guarantee the uniqueness of the payout instruction and prevent occurrences of unintended double payments.
payerPaymentReference	String	Mandatory	1-35 Alphanumeric characters. A Merchant specific reference. This reference could for example be order id or similar. The property is not used by Swish but is included in responses back to the client.
payerAlias	String	Mandatory	Numeric, 10-11 digits The Swish number of the merchant that makes the payout payment. **Note**: MSS only knows about the provided signing certificate, thus this property must be set to the swish number of that certificate or the request will be rejected due to mismatch with the signing certificate Common Name property.
payeeAlias	String	Mandatory	The mobile phone number of the person that receives the payment.

payeeSsn	String	Mandatory	YYYYMMDDXXXX The Social Security Number of the person that receives the payment.
amount	String	Mandatory	Example: 100.05 The amount to be paid. Note that only period/dot (”.”) are accepted as decimal character with maximal 2 digits after. Digits after separator are optional.
currency	String	Mandatory	The only supported value is: SEK The currency to use.
payoutType	String	Mandatory	Only supported value is: PAYOUT Immediate payout.
message	String	Optional	0-50 Alphanumeric characters. Custom message. **Note**: For MSS, an error simulation code can be set in this property in order to simulate an error situation. Refer to section 9.5 for available codes.
instructionDate	String	Mandatory	YYYY-MM-DDTHH:MM:SS Date and time for when the payout instruction was supplied. Example: 2019-12-03T11:07:16
signingCertificateSerialNumber	String	Mandatory	Serial number of the signing certificate in hexadecimal format (without any leading ‘0x’ characters). The public key of the certificate with this serial number will be used to verify the signature. **Note**: MSS only knows about the provided signing certificate, thus this property must be set to the serial number of that certificate or MSS will reject the request due to failed signature validation.

callbackUrl	String	Optional	https://<host[:port]>/... URL that Swish system will use to notify caller about the result of the payment request. The URL **must** use HTTPS. If not set (or not provided in the payload) it is the responsibility of the caller to check the status of the request using GET operation as described in chapter 10.
signature	String	Mandatory	Base64 encoded. Signature of the hashed payload. **Note**: MSS requires that the payload is signed using the private key of the provided signing certificate.

11.2 How to create the payload signature

This chapter provides information on the process of creating the payload signature. Note that no details are provided since tools that users have at their disposal vary.
Before creating the signature it is crucial that the serial number of the *signing certificate* is included in the payload (property `signingCertificateSerialNumber`). The serial number is required in order for the system to be able validate the integrity of the payload as well as the validity of the signing certificate itself.

The process of creating a payout request JSON string to include in the request body could typically look like this:

1. Unless already known, extract the serial number from the signing certificate.
   `"7D70445EC8EF4D1E3A713427E973D097"`

2. Create a JSON formatted payload string representing the payout instruction.

```
{ "amount": "100.00",
  "currency": "SEK",
  "instructionDate": "2019-12-02T12:41:20",
  "message": "example message",
  "payeeAlias": "0728648607",
  "payeeSSN": "196702234292",
  "payerAlias": "1234679304",
  "payerPaymentReference": "mockedPayerPaymentReference",
  "payoutInstructionUUID": "822735C159A944DBB2B3546717BC919F",
  "payoutType": "PAYOUT",
  "signingCertificateSerialNumber": "7D70445EC8EF4D1E3A713427E973D097"
}
```

3. Hash the payload string (payout instruction) using SHA512 algorithm.

4. Sign the hashed payload string using RSA algorithm and the signing certificate private key.

5. Convert the signing result to Base64 format for inclusion in the request as `signature` value.
   `"IUybofWGga+uS2+APequDO5kh..."`

6. Create the complete payout request JSON string using the result from step 2 and 5.

```
{    "payload":
       { "amount": "100.00",
         "currency": "SEK",
         "instructionDate": "2019-12-02T12:41:20",
         "message": "example message",
         "payeeAlias": "0728648607",
         "payeeSSN": "196702234292",
         "payerAlias": "1234679304",
         "payerPaymentReference": "mockedPayerPaymentReference",
         "payoutInstructionUUID": "822735C159A944DBB2B3546717BC919F",
         "payoutType": "PAYOUT",
         "signingCertificateSerialNumber": "7D70445EC8EF4D1E3A713427E973D097"
    },
    "callbackUrl": "https://myfavoritesite.dummy.domain",
    "signature":"IUybofWGga+uS2+APequDO5kh..."
}
```

Important: Take care that the sent payload property value is ***exactly*** as it was when it was hashed. Otherwise the signature validation in MSS will most likely fail and the request rejected.

Tip: Remove all white spaces and newlines from the payload value *before* hashing (beware that the `message` property might contain spaces that should be preserved).

11.3 HTTP status codes

The following HTTP status codes could be received in a create payout response:

HTTP status codes	Returned scenarios
201 Created	Returned when Payout was successfully created. Will include a `Location` header that links to the created payout instruction. *Note that the message body is empty*.
4xx	Request failed for some reason, payout instruction processing has not been initiated. Details are provided in the message body. The body contains a a list of JSON formatted error objects. Example body: `[{"errorCode":"RP02","errorMessage":"Invalid format 'message'","additionalInformation":null}]` Example status codes: 401 Unauthorized: Returned when there are problems with the authentication certificate or the payout signature property. 422 Unprocessable Entity: Returned when there are data validation errors, e.g. payerSsn validation failed.
500 Internal Server Error	A server error has occurred, payout has not been initiated.

11.4 Error simulation codes

The API user can trigger (simulate) an error situation by setting the `message` property of the *Create payout request* payload to an appropriate value as indicated in the following table.

Note that the codes listed here are only those that can be simulated using the payload `message` property. These error codes are just a subset of all possible error codes that could be returned by the system, for example, providing a wrongly formatted `payeeSsn` property in the payload would result in `errorCode` PA06.

Simulate errors (first synchronous response):

Error codes	Description
PA01	Invalid format of a field or otherwise invalid information in request
ACMT13	Bank does not support ’PAYOUT’.
ACMT14	Payer is not allowed to perform ’PAYOUT’.
ACMT15	Payee is not allowed to receive ’PAYOUT’
TM01	Swish system timed out.
RF07	Transaction could not be executed.

Simulate delayed errors (errors provided in callback):

This is not yet supported by MSS.

12 Retrieve payout result (GET)

The client can retrieve payout result information for an initiated payout request by sending a GET request to the following URL:

`https://mss.cpc.getswish.net/swish-cpcapi/api/v1/payouts/<payoutInstructionUUID>`

The complete URL is available in the HTTP `Location` header returned in the response message for a previous successful *create payout request* call.

If the previous *create payout request* call simulated a delayed error the response of the GET operation will have a `status` property with value ERROR and properties `errorCode` and `errorMessage` will be set accordingly. See chapter 9.5 for possible errors that can be simulated with a delay.

12.1 Example, Retrieve payout result

Example on what the result from a GET operation could look like. In this example the GET request is performed 3 times with a delay (compare with chapter 9) to give time for the payout instruction status to change.

1. GET request sent within 4 seconds from when the POST request was sent.
2. GET request sent within 8 seconds from when the POST request was sent.
3. GET request sent after 8 seconds from when the POST request was sent.

```
1 Retrieve payout result (within 4 seconds, status CREATED):
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --
cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type:
application/json" https://mss.cpc.getswish.net/swish-
cpcapi/api/v1/payouts/AEFCF44F762E485C87DA41230861B79A

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Wed, 04 Dec 2019 12:57:01 GMT

{"paymentReference":null,"payoutInstructionUUID":"AEFCF44F762E485C87DA41230861B79A"
,"payerPaymentReference":"MTS-DUMMY-PAYMENT-
REF","callbackUrl":"https://myfavoritesite.dummy.domain/callback/","payerAlias":"12
34679304","payeeAlias":"46722334455","payeeSSN":"197501088327","amount":1.00,"curre
ncy":"SEK","message":"Tieto Test
Message","payoutType":"PAYOUT","status":"CREATED","dateCreated":"2019-12-
04T12:56:59.874","datePaid":null,"errorMessage":null,"additionalInformation":null,"
errorCode":null}
```

2 Retrieve payout result (after 4 but within 8 seconds, status DEBITED):

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type: application/json" https://mss.cpc.getswish.net/swish-cpcapi/api/v1/payouts/AEFCF44F762E485C87DA41230861B79A

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Wed, 04 Dec 2019 12:57:05 GMT

{"paymentReference":"78919BD775D6442FB2CCFAAEAEDC9598","payoutInstructionUUID":"AEFCF44F762E485C87DA41230861B79A","payerPaymentReference":"MTS-DUMMY-PAYMENT-REF","callbackUrl":"https://myfavoritesite.dummy.domain/callback/","payerAlias":"1234679304","payeeAlias":"46722334455","payeeSSN":"197501088327","amount":1.00,"currency":"SEK","message":"Tieto Test Message","payoutType":"PAYOUT","status":"DEBITED","dateCreated":"2019-12-04T12:56:59.874","datePaid":"2019-12-04T12:57:03.875","errorMessage":null,"additionalInformation":null,"errorCode":null}
```

3 Retrieve payout result (after 8 seconds, status PAID):

```
$ curl -s -S -i --cert ./Swish_Merchant_TestCertificate_1234679304.p12:swish --cert-type p12 --cacert ./Swish_TLS_RootCA.pem --tlsv1.2 --header "Content-Type: application/json" https://mss.cpc.getswish.net/swish-cpcapi/api/v1/payouts/AEFCF44F762E485C87DA41230861B79A

HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Wed, 04 Dec 2019 12:57:11 GMT

{"paymentReference":"78919BD775D6442FB2CCFAAEAEDC9598","payoutInstructionUUID":"AEFCF44F762E485C87DA41230861B79A","payerPaymentReference":"MTS-DUMMY-PAYMENT-REF","callbackUrl":"https://myfavoritesite.dummy.domain/callback/","payerAlias":"1234679304","payeeAlias":"46722334455","payeeSSN":"197501088327","amount":1.00,"currency":"SEK","message":"Tieto Test Message","payoutType":"PAYOUT","status":"PAID","dateCreated":"2019-12-04T12:56:59.874","datePaid":"2019-12-04T12:57:03.875","errorMessage":null,"additionalInformation":null,"errorCode":null}
```

12.2 HTTP status codes

The following HTTP status codes could be received in a retrieve payout result response:

HTTP status codes	Returned scenarios
200 OK	Returned when payout instruction was found, payout instruction details provided in the message body.
4xx	Request failed for some reason, for example the payoutInstructionUUID does not exist.
500 Internal Server Error	Returned if there was some unknown/unforeseen error that occurred on the server.

13 Tips on how to trigger an error

In addition to using the `message` property for simulating errors in the synchronous and asynchronous responses other methods can be used. Some examples are described below:

- HTTP communication
  - Provide an incorrect address e.g. i.e. remove “s” in paymentrequests – HTTP 404 Not Found
  - Remove client certificate – “Received fatal alert: handshake_failure”
- Payment Request
  - "payeePaymentReference"
    - Provide too long – “FF08","errorMessage":"Payment Reference is invalid"
    - Provide NULL – “FF08","errorMessage":"Payment Reference is invalid"
  - "amount"
    - Provide “,” e.g. 12,09 – “PA02","errorMessage":"Amount value is missing or not a valid number”
    - Provide less than 1 e.g. 0.5 - "AM06","errorMessage":"Specified transaction amount is less than agreed minimum"
    - Provide 3 decimals e.g. 100.777 – “PA02","errorMessage":"Amount value is missing or not a valid number”
  - "payeeAlias"
    - ~~Provide a number that does not match the value in the certificate – “PA01","errorMessage":"Parameter is not correct."~~
      Not for MSS!
  - "payerAlias"
    - Provide a too long or short number - "BE18","errorMessage":"Payer alias is invalid"
  - "currency"
    - Provide another value than “SEK” - :"AM03","errorMessage":"Invalid or missing Currency"
- Retrieve payment/refund result (callback confirmation)
  - Provide an invalid ID - HTTP/1.1 404 Not Found
- Refund
  - "payerPaymentReference"
    - Provide too long reference – “FF08","errorMessage":"Payment Reference is invalid"
    - Provide NULL – “FF08","errorMessage":"Payment Reference is invalid"
  - "originalPaymentReference"
    - This value is taken from the Payment Request callback element “paymentReference”
    - Use a value that is not valid i.e. change a value - "RF02","errorMessage":"Original Payment not found or original payment is more than than 13 months old"
  - "payerAlias"
    - ~~Provide a number that does not match the value in the certificate – “PA01","errorMessage":"Parameter is not correct."~~
      Not for MSS!
  - "amount"
    - Provide an amount that is greater than the original payment - "RF08","errorMessage":"Amount value is too large or amount exceeds the amount of the original payment minus any previous refunds"
      Not for MSS!
    - Provide “,” e.g. 12,09 – “PA02","errorMessage":"Amount value is missing or not a valid number”
    - Provide less than 1 e.g. 0.5 - "AM06","errorMessage":"Specified transaction amount is less than agreed minimum"
    - Provide 3 decimals e.g. 100.777 – “PA02","errorMessage":"Amount value is missing or not a valid number”
  - "currency"

- Provide another value than “SEK” - :"AM03","errorMessage":"Invalid or missing Currency"